

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of :  
Michiko MATSUMOTO et al. :  
Serial No. NEW : **Attn: APPLICATION BRANCH**  
Filed November 14, 2003 : Attorney Docket No. 2003-1649A

APPARATUS, METHOD AND PROGRAM  
FOR CONTENTION ARBITRATION

---

**CLAIM OF PRIORITY UNDER 35 USC 119**

Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Sir:

Applicants in the above-entitled application hereby claim the date of priority under the International Convention of Japanese Patent Application No. 2002-331905, filed November 15, 2002, as acknowledged in the Declaration of this application.

A certified copy of said Japanese Patent Application is submitted herewith.

Respectfully submitted,

Michiko MATSUMOTO et al.

By 

Nils E. Pedersen  
Registration No. 33,145  
Attorney for Applicants

NEP/krq  
Washington, D.C. 20006-1021  
Telephone (202) 721-8200  
Facsimile (202) 721-8250  
November 14, 2003

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日                      2 0 0 2 年 1 1 月 1 5 日  
Date of Application:

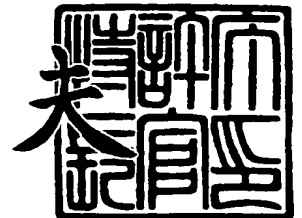
出 願 番 号                      特 願 2 0 0 2 - 3 3 1 9 0 5  
Application Number:  
[ST. 10/C] :                      [ J P 2 0 0 2 - 3 3 1 9 0 5 ]

出      願      人                      松 下 電 器 産 業 株 式 有 限 公 司  
Applicant(s):

2 0 0 3 年    9 月    1 日

特許庁長官  
Commissioner,  
Japan Patent Office

今 井 康



出証番号    出証特 2 0 0 3 - 3 0 7 1 0 0 0

【書類名】 特許願

【整理番号】 2037340046

【提出日】 平成14年11月15日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 12/00 572  
G06F 15/177 682  
G06F 15/177 678  
G06F 12/00 571

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 松下電器産業株式会社内

【氏名】 松本 美智子

【発明者】

【住所又は居所】 大阪府門真市大字門真 1 0 0 6 番地 株式会社松下ソフトリサーチ内

【氏名】 鈴木 良章

【特許出願人】

【識別番号】 000005821

【氏名又は名称】 松下電器産業株式会社

【代理人】

【識別番号】 100097445

【弁理士】

【氏名又は名称】 岩橋 文雄

【選任した代理人】

【識別番号】 100103355

【弁理士】

【氏名又は名称】 坂口 智康

**【選任した代理人】****【識別番号】** 100109667**【弁理士】****【氏名又は名称】** 内藤 浩樹**【手数料の表示】****【予納台帳番号】** 011305**【納付金額】** 21,000円**【提出物件の目録】****【物件名】** 明細書 1**【物件名】** 図面 1**【物件名】** 要約書 1**【包括委任状番号】** 9809938

【書類名】 明細書

【発明の名称】 競合調停装置および競合調停方法

【特許請求の範囲】

【請求項 1】 複数のアプリケーションがアクセス対象に同時にアクセス要求をしたときに起こるアクセス競合を調停する装置において、

前記アクセス対象が必要とする一つ以上の競合発生要因を表すリソースごとに、現在アクセス可能なアプリケーションがどれかを示すリソース情報を保持するリソース情報記憶部と、

第 1 のアプリケーションから前記アクセス対象へのアクセス要求を受けた際に、前記リソース情報記憶部から前記リソース情報を取得し、現在アクセス権を持っている第 2 のアプリケーションが存在する場合は前記第 1 のアプリケーションと前記第 2 のアプリケーションのどちらにアクセス権を持たせるかを判定するリソースアクセス判定部と、

前記第 1 のアプリケーションが、前記アクセス対象が必要とする全ての前記リソースのアクセス権を取得したかを判定するアクセス対象アクセス判定部とを備えた競合調停装置。

【請求項 2】 前記第 1 のアプリケーションからアクセス要求を受信するアプリケーション I F 部を新たに持つ請求項 1 に記載の競合調停装置。

【請求項 3】 前記アクセス対象が論理デバイスの場合、前記アクセス対象アクセス判定部において、前記第 1 のアプリケーションが前記論理デバイスにアクセス可能な場合は前記論理デバイスを制御するデバイスドライバを呼び出す実行部を新たに持つ請求項 1 に記載の競合調停装置。

【請求項 4】 前記実行部において、前記第 1 のアプリケーションが前記論理デバイスへのアクセス権を取得できなかった場合に、前記アプリケーション I F 部は前記第 1 のアプリケーションにエラーを通知する請求項 3 に記載の競合調停装置。

【請求項 5】 前記論理デバイスが必要とするリソースとを対応付けしたデバイス情報を保持し、前記第 1 のアプリケーションからのアクセス要求を受信したときに必要なリソースを前記デバイス情報から取得する使用リソース取得部を新

たに備えた請求項 1 に記載の競合調停装置。

【請求項 6】 前記リソースアクセス判定部において、アクセス競合が発生した場合にアプリケーションの優先度に基づいてアクセス権を取得させるアプリケーションを判定する請求項 1 に記載の競合調停装置。

【請求項 7】 前記アプリケーションの優先度の情報を保持するアプリケーション情報記憶部を新たに持つ請求項 6 に記載の競合調停装置。

【請求項 8】 前記アプリケーションの優先度が同じ場合に、先にアクセス要求をしたアプリケーションにアクセス権を取得させるか、後からアクセス要求をしたアプリケーションにアクセス権を取得させるかの情報をリソースごとにリソース情報に持たせた請求項 6 に記載の競合調停装置。

【請求項 9】 アプリケーションから最初の要求であるアクセス開始要求を受信したときに競合調停をおこない、アクセス権を取得したかどうかの情報を前記アプリケーション情報に持たせ、前記アプリケーションが 2 回目以降にアクセス要求したときは、前記アプリケーション情報を参照してアクセス可能かどうかを判定する請求項 7 に記載の競合調停装置。

【請求項 10】 前記第 2 のアプリケーションは前記第 1 のアプリケーションにアクセス権を奪われた場合に、前記第 2 のアプリケーションから再度アクセス要求があったときに前記第 2 のアプリケーションにエラーを通知する請求項 9 に記載の競合調停装置。

【請求項 11】 前記第 1 のアプリケーションが最後にアクセス終了要求をしたときに、前記第 1 のアプリケーションにアクセス権を奪われていたために今までデバイスにアクセスできなかった第 3 のアプリケーションをアクセス可能とし、前記デバイスを前記第 3 のアプリケーションがアクセスするための状態に復元させる制御命令をデバイスドライバに通知する請求項 9 に記載の競合調停装置。

【請求項 12】 前記リソース情報記憶部は、多重アクセスを許すデバイスの場合はその該当機能のリソース情報を複数保持する請求項 1 に記載の競合調停装置。

【請求項 13】 前記リソース情報記憶部は、ある条件に基づいて多重アクセスを許すデバイスの場合は、その該当機能のリソース情報を保持する請求項 1 に

記載の競合調停装置。

【請求項 14】 一つのデバイスに対して複数のアプリケーションがアクセス要求をしたときに起こるアクセス競合を調停する方法において、

第1のアプリケーションからデバイスへのアクセス要求を受信する工程と、  
アプリケーションのアクセス対象デバイスとそのデバイスが使用する機能を表すリソースとを対応付けしたデバイス情報を予め保持しており、前記第1のアプリケーションがアクセスする前記デバイスが使用するリソースを前記デバイス情報から取得する工程と、

リソースに対して現在アクセス権を持っているアプリケーションが存在するか、存在する場合はどのアプリケーションかを示す更新可能なリソース情報を保持し、必要に応じて更新する工程と、

前記リソース情報から前記第1のアプリケーションがアクセス対象としている前記デバイスが使用する前記リソースのリソース情報を取得し、現在アクセス権を持っている第2のアプリケーションとアクセス要求をしている前記第1のアプリケーションとで、どちらにアクセス権を持たせるかを判定する工程と、

アクセス対象の前記デバイスが使用する全ての前記リソースに対して、前記第1のアプリケーションがアクセス権を取得し、前記デバイスにアクセス可能かを判定する工程と、

デバイスにアクセス可能な場合は前記デバイスを制御するデバイスドライバを呼び出す工程とを備えた競合調停方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、複数のアプリケーションが起動するコンピュータシステム内において、複数のアプリケーションがアクセス対象に同時にアクセスする時に生じる競合の調停方法に関するものである。

【0002】

【従来の技術】

複数のアプリケーションが起動するコンピュータシステム内では、複数のアプ

リケーションが一つのデバイスに同時にアクセス要求をすることにより競合が発生する。共有資源の排他制御方法としては、複数の計算機からその共有資源へのアクセス要求が来たときに、排他制御識別子を用いて競合調停をする方法が開示されている（例えば特許文献 1 参照）。

#### 【0003】

また一つのコンピュータシステム内において同時に複数のスレッドから記憶データにアクセス要求が来たときに、先勝ちにより競合調停をする方法も開示されている（例えば特許文献 2 参照）。

#### 【0004】

また一つのコンピュータシステム内において、調停機能を持たせて他のデバイスを制御する制御デバイスを設け、さらにアクセス要求に付与した優先度に基づいて競合調停をする方法も開示されている（例えば特許文献 3 参照）。

#### 【0005】

また一つの制御機器に対して複数の制御命令が来たときに、所定の条件に基づいて競合調停をする方法も開示されている（例えば特許文献 4 参照）。

#### 【0006】

##### 【特許文献 1】

特開 2002-175287 号公報

##### 【特許文献 2】

特開平 10-187527 号公報

##### 【特許文献 3】

特開 2000-231458 号公報

##### 【特許文献 4】

特開 2001-346276 号公報

#### 【0007】

##### 【発明が解決しようとする課題】

しかし、このような従来の技術で、例えば一つのコンピュータシステム内で競合調停を行う場合では、競合調停を実際に存在するデバイス単位で行っていた。そのため実際にはデバイスは一つしかないが、ある条件の下では複数のアプリケ



ーションから同時にアクセスが可能である、またアプリケーションがある数までなら同時にアクセスが可能である、といった多重アクセスを許すデバイスにおいて競合調停を行っても、一つのアクセス要求しか受け付けず、デバイスの特性を十分に活かすことができなかった。また他のデバイスと配線結合され、単体では I/O ポートを持たないデバイスが複数のデバイス間で共有されるような場合では、アプリケーション間で状態遷移表を用いて競合を回避するしかなく、アプリケーションやデバイスの構成に依存せざるを得ず、数が増えるに従い調停が困難であった。

#### 【0008】

本発明は、このようなデバイスの特性を活かしきるために提案された競合調停方法である。アクセス対象とするデバイスと、実際に競合調停を行うリソースとを分離して考えることにより、実際には一つしか存在しないが多重アクセスを許す機能を持つデバイスに対して、その機能を活かしきる競合調停を実現可能とすることを目的とする。

#### 【0009】

##### 【課題を解決するための手段】

上記の目的を達成するために、一つのデバイスに対して複数のアプリケーションがアクセス要求をしたときに起こるアクセス競合を調停する装置において、第1のアプリケーションから第1のデバイスへのアクセス要求を受信するアクセス要求受信部と、アプリケーションのアクセス対象デバイスとそのデバイスが使用する機能を表すリソースとを対応付けしたデバイス情報を予め保持しており、前記第1のアプリケーションがアクセスする前記第1のデバイスが使用する第1のリソースを前記デバイス情報から取得する使用リソース取得部と、リソースに対して現在アクセス権を持っているアプリケーションが存在するか、存在する場合はどのアプリケーションかを示す更新可能なリソース情報を保持するリソース情報記憶部と、前記リソース情報記憶部から前記第1のアプリケーションがアクセス対象としている前記第1のデバイスが使用する前記第1のリソースのリソース情報を取得し、現在アクセス権を持っている第2のアプリケーションとアクセス要求をしている前記第1のアプリケーションとで、どちらにアクセス権を持たせ

るかを判定するリソースアクセス判定部と、アクセス対象の前記第 1 のデバイスが使用する全ての前記第 1 のリソースに対して、前記第 1 のアプリケーションがアクセス権を取得し、前記デバイスにアクセス可能かを判定するアクセス対象アクセス判定部と、デバイスにアクセス可能な場合は前記第 1 のデバイスを制御するデバイスドライバを呼び出す実行部とを備えた競合調停装置である。

#### 【0010】

##### 【発明の実施の形態】

以下、本発明の実施の形態について、図面を参照しながら説明する。図 1 は、本発明の実施の形態における競合調停装置の構成の一例を示すブロック図である。図 1 において、競合調停装置は、アプリケーション I/F 部 11 と、アプリケーション情報記憶部 12 と、使用リソース取得部 13 と、リソース情報記録部 14 と、リソースアクセス判定部 15 と、アクセス対象アクセス判定部 16 と、実行部 17 とを備え、アプリケーションから受け取った要求に基づいて競合調停を処理し、対応するデバイスを使用するためにデバイスドライバの呼び出しを行う。

#### 【0011】

アプリケーション I/F 部 11 は、アプリケーションからアクセス開始要求を受信すると、使用リソース取得部 13 にアクセス対象論理デバイスへのアクセス可否判定を要求し、アクセスを要求したアプリケーションの優先度をアプリケーション情報記憶部 12 に格納処理する。また、アプリケーションからアクセス要求を受信すると、実行部 17 を呼び出すことにより、アプリケーションがデバイスへのアクセスを行う。なお、アプリケーションがデバイスへのアクセス不可の場合は、実行部 17 から受信したエラーをアプリケーションに通知する。更に、アプリケーションからアクセス終了要求を受信すると、使用リソース取得部 13 にアプリケーションが保持していたアクセス権の解放を要求する。

#### 【0012】

アプリケーション情報記憶部 12 は、アプリケーション毎にアプリケーション情報を格納している。アプリケーション情報は、アプリケーション I/F 部 11 が設定するアプリケーションの優先度と、アクセス対象の論理デバイス名と、実行部 17 が設定する論理デバイスにアクセス可能か不可かの情報から構成されてい

る。

#### 【0013】

使用リソース取得部 13 は、論理デバイスと前記論理デバイスが必要とするリソースとの関係を示したデバイス情報を保持しており、アプリケーション I F 部 11 からアクセス開始要求が通知されると、前記デバイス情報を用いて、アクセス開始が要求されている論理デバイスが必要とするリソースを検出し、検出したリソースをリソースアクセス判定部 15 に通知する。なお、1つの論理デバイスに対して複数のリソースが関係付けられてもよく、リソースアクセス判定部 15 には、複数のリソースを同時に通知してもよいし、1つずつ順番に通知してもよい。また、論理デバイスとリソースとの関係については、図 2 を用いて後に説明する。

#### 【0014】

リソース情報記憶部 14 は、全てのリソースに対して既にアクセス権を取得しているアプリケーションが存在するか否か、また、存在する場合は、アクセス権を取得しているアプリケーションを特定する情報をリソース情報として格納している。このリソース情報は、リソースアクセス判定部 15 が、リソースへのアクセス権を保持するアプリケーションが変更になったと判断した場合に、更新設定を行う。

#### 【0015】

リソースアクセス判定部 15 は、使用リソース取得部 13 から通知されたリソースに対するリソース情報をリソース情報記憶部 14 から取得し、既にアクセス権を取得しているアプリケーションが存在する場合は、そのアプリケーションとアクセス開始を要求しているアプリケーションとでどちらが優先かを、アプリケーション情報記憶部 12 に格納してあるアプリケーションの優先度に基づいて判定する。判定結果として、アクセス権を取得するアプリケーションが変更になった場合は、リソース情報記憶部 14 の対応するリソースのリソース情報を更新する。

#### 【0016】

アクセス対象アクセス判定部 16 は、使用リソース取得部 13 が検出したリソ

ースと、リソース情報記憶部 14 から取得したリソース情報を用いて、アクセス開始を要求しているアプリケーションがアクセスしたい論理デバイスが使用する全てのリソースに対してアクセス権を取得しているか否かを判定し、全てのリソースに対してアクセス権を取得している場合に、論理デバイスへのアクセスが可能であると決定する。

#### 【0017】

実行部 17 は、アクセス対象アクセス判定部 16 から論理デバイスへのアクセスが可能か否かの情報を受け取り、受け取った情報に基づいて、アプリケーション情報記憶部 12 に格納されているアプリケーション情報の更新を行う。また、アプリケーション I/F 部 11 からアクセス要求が通知されると、アプリケーション情報記憶部 12 に格納されているアクセスを要求したアプリケーションがアクセス可能な場合は、対応するデバイスドライバを呼び出す。なお、アプリケーションがデバイスへのアクセス不可の場合は、実行部 17 は、アプリケーション I/F 部 11 にエラーを通知する。

#### 【0018】

本発明ではリソースを物理デバイスが提供する機能と捉え、リソース単位で競合調停を行う。図 2 はアプリケーションのアクセス対象の論理デバイスとリソースと物理デバイスとの関係を示す構成図の一例である。図 2 では I/O ポートを持った物理デバイスは S D S P 物理デバイスと M I D I 物理デバイスであり、スピーカー物理デバイスは S D S P 物理デバイスと M I D I 物理デバイスに配線されており、I/O ポートは持っていないため、スピーカー物理デバイスのみを制御するということはできない。さらにスピーカー物理デバイスは S D S P 物理デバイスと M I D I 物理デバイスとで共有されているため、アプリケーション A P 1 が S D S P 物理デバイスにアクセスするときは、アプリケーション A P 2 は M I D I 物理デバイスにはアクセスできず、また逆も起こりえる。このように S D S P 物理デバイスと M I D I 物理デバイスに対して同時にアクセス要求がある場合は、アプリケーションからは見えないスピーカー物理デバイスに対してアクセス競合が発生する。そのためこのアクセス競合を回避するには、従来ではアプリケーション間で調停を行う必要があり、アプリケーションやデバイスの種類が増

えるに従い、調停が困難になる。そこで、本発明では先に述べたとおり、物理デバイスが提供する機能をリソースと捉え、そのリソース単位で競合調停を行う。

#### 【0019】

図2ではSDSP物理デバイスの機能を表すSDSPリソースと、MIDI物理デバイスの機能を表すMIDIリソースと、スピーカー物理デバイスの機能を表すスピーカリソースが存在する。またアプリケーションがアクセス対象とするのは論理デバイスである。論理デバイスは実際に使用する物理デバイスの機能、つまりリソースを必要とする。図2ではSDSP論理デバイスはSDSPリソースとスピーカリソースを必要とし、MIDI論理デバイスはMIDIリソースとスピーカリソースを必要とする。そしてアプリケーションがアクセス対象とする論理デバイスにアクセス可能となるのは、その論理デバイスが必要とする全てのリソースに対してアクセス権が得られたときである。また本発明では個々のリソースに対してアクセス権が得られるかどうかは、先勝ちや後勝ちなどの決め方以外に、アプリケーションの優先度に応じて決めることができる。例えば図2において、アプリケーションAP1がSDSP論理デバイスにアクセス要求をし、アプリケーションAP2がMIDI論理デバイスにアクセス要求をした場合を考える。AP1とAP2ではAP2の方が優先度が高いとする。この場合、SDSPリソースにはAP1しかアクセス要求をしていないので、AP1がSDSPリソースのアクセス権を取得する。また同様にMIDIリソースにはAP2しかアクセス要求をしていないので、AP2がMIDIリソースのアクセス権を取得する。スピーカリソースについては、AP1とAP2の両方のアクセス対象論理デバイスが必要としているため、競合が発生する。ここではAP2の方が優先度が高いため、AP2がスピーカリソースのアクセス権を取得できる。そしてAP2はアクセス対象のMIDI論理デバイスが必要とするリソース全てにおいてアクセス権が取得できたため、アクセスが可能となる。またAP1はアクセス対象のSDSP論理デバイスが必要とするリソースのうち、スピーカリソースについてアクセス権を取得できなかったため、アクセス不可となる。

#### 【0020】

図3にアプリケーションのアクセス対象論理デバイスとリソースと物理デバイ

スとの構成の別の例を示す。図3は、物理デバイスとしては1回線しか存在しないが、実際には同時に3回線まで使用できるマルチコール対応の回線デバイスの場合を示している。この場合、アプリケーションのアクセス対象である論理デバイスおよび回線物理デバイスは1つしかないが、回線リソースは3つ存在する、と考える。そして回線論理デバイスは1つの回線リソースを必要とする。例えば図3においてアプリケーションAP1、AP2、AP3、AP4がともに回線論理デバイスに対してアクセス要求をし、優先度は $AP1 > AP2 > AP3 > AP4$ の順でAP1が最も高い場合を考える。まずAP1が回線論理デバイスに対してアクセス要求をすると、AP1は3つある回線リソースのうち、1つ目の回線リソース32のアクセス権を取得する。そして次にAP2が回線論理デバイスにアクセス要求をすると、AP2は2つ目の回線リソース33のアクセス権を取得する。次に最も優先度の低いAP4が回線論理デバイスにアクセス要求をすると、AP3は3つ目の回線リソース34のアクセス権を取得する。そこへAP4より優先度の高いAP3が回線論理デバイスにアクセス要求をすると、AP4とAP3とで優先度を比較し、AP3の方が高いため、AP4は3つ目の回線リソース34に対してアクセス不可となる。またAP3は回線リソース34へのアクセス権を取得する。そして、AP1、AP2、AP3が回線論理デバイスにアクセス可能となる。

#### 【0021】

また図4に別の構成例を示す。図4は、物理デバイスとしては1つしか存在しないが、実際には同じコーデックであれば録音機能と再生機能は同時に使用できるSDSPデバイスの場合を示している。この場合、アプリケーションのアクセス対象論理デバイスはSDSP録音論理デバイスとSDSP再生論理デバイスの2つが存在し、リソースはコーデック機能を表すSDSPコーデックリソースと、録音機能を表すSDSP録音リソースと、再生機能を表すSDSP再生リソースの3つが存在する、と考える。さらにSDSPコーデックリソースはコーデックが同じ場合は録音機能と再生機能を同時に使用できるため、2つ存在する、と考えるのではなく、コーデック情報を付加情報として持たせ、1つのみ存在する、と考える。そしてアクセス要求をしているアプリケーションが複数存在する場

合は、優先度が最も高いアプリケーションのコーデックを設定し、それと同じコーデックであればSDSPコーデックリソースにはアクセス可能とする。例えば図4においてアプリケーションAP1、AP2がSDSP録音論理デバイスにアクセス要求をし、AP3がSDSP再生論理デバイスにアクセス要求をし、優先度は $AP1 > AP2 > AP3$ の順でAP1が最も高く、またAP1、AP2、AP3はともに同じコーデックXでアクセス要求をする場合を考える。まずAP1がSDSP録音論理デバイスに対してアクセス要求をするとAP1はSDSPコーデックリソースのアクセス権を取得し、Xに設定する。またSDSP録音リソースに対してもアクセス権を取得する。次にAP2がSDSP録音論理デバイスにアクセス要求をすると、AP2はSDSPコーデックリソースに対しては、より優先度の高いAP1と同じコーデックのためアクセス権を取得できるが、SDSP録音リソースについてはAP1と競合するため、アクセス権を取得できない。次にAP3がSDSP再生論理デバイスにアクセス要求をすると、AP3はSDSPコーデックリソースに対しては、より優先度の高いAP1と同じコーデックであるためアクセス権を取得でき、またSDSP再生リソースに対しては競合が発生していないのでアクセス権を取得できる。つまりAP1、AP3はともにアクセス対象論理デバイスが必要とする全てのリソースにアクセス可能なため、それぞれSDSP録音論理デバイスおよびSDSP再生論理デバイスにアクセス可能であるが、AP2はSDSP録音リソースのアクセス権が取得できないため、SDSP録音論理デバイスにアクセス不可となる。

#### 【0022】

それでは次に本発明の詳細な処理手順を説明する。アプリケーションはまず始めにアクセス開始要求をし、デバイスにアクセス可能かどうかを判定して、可能な場合はデバイスに必要な設定等を行う。そして実際にデバイスにアクセスするときはアクセス要求をする。そして最後にアプリケーションが終了するなどで、デバイスへのアクセスも終了するときに、アクセス終了要求をし、デバイスの状態を初期化したり、場合によっては他のアプリケーションが使用するための設定をおこなったりする。

#### 【0023】

図5にアプリケーションのアクセス開始要求時の処理手順を示す。まずアプリケーション I F 部 11 は、アプリケーション A P 1 からアクセス対象論理デバイス D E V 1 へのアクセス開始要求と A P 1 の優先度を受信する (S T 1)。次にアプリケーション I F 部 11 は、A P 1 のアプリケーション情報をアプリケーション情報記憶部 12 に登録する (S T 2)。アプリケーション情報には、アプリケーションの優先度、アクセス対象論理デバイス名、その論理デバイスにアクセス可能か不可かのフラグなどがあるが、このとき登録するアプリケーション情報はアプリケーションの優先度、アクセス対象論理デバイス名である。次にアプリケーション I F 部 11 は、A P 1 から D E V 1 へのアクセス開始要求がきたことを使用リソース取得部 13 に通知する。そして使用リソース取得部 13 は D E V 1 が使用するリソースの I D を取得する (S T 3)。これは使用リソース取得部 13 が予め保持しているアクセス対象論理デバイスとそれが必要とするリソース I D との関係を示したデバイス情報から取得する。そして使用リソース取得部 13 は、取得したリソース I D をリソースアクセス判定部 15 に通知する。そしてリソースアクセス判定部 15 はそのリソースのリソース情報をリソース情報記憶部 14 から取得する (S T 3)。このリソース情報はリソースごとに存在し、現在アクセス権を取得しているアプリケーションを示している。アクセス権を取得しているアプリケーションが存在しない場合は N U L L を示している。また図3に示した例のように同一リソースが複数存在する場合は、同一リソース I D のリソース情報が複数存在する。次にリソースアクセス判定部 15 はそのリソースに対して A P 1 がアクセス権を取得できるかを判定する (S T 4)。この処理の詳細なフローチャートを図6に示す。まず図3に示した例のように同一リソースが複数存在するかを、リソース情報記憶部 14 を参照して、そのリソース I D のリソース情報が複数あるか調べることで判定する (S T 11)。存在する場合、つまり図3の例のような場合では、まず全てのリソースに対して、アクセス権を持っているアプリケーションが存在するかを、リソース情報記憶部 14 に保持されているリソース情報を用いて調べる (S T 12)。そして全てのリソースに対して存在する場合は、アプリケーション情報記憶部 12 に保持されているアプリケーション情報を参照して最も優先度の低いアプリケーション A P 2 を検索する (



ST13)。そしてAP1とAP2とでアプリケーション情報に設定されている優先度を比較し(ST14)、もしAP1の優先度が高い場合(ST15)はそのリソース情報に対して、アクセス権を取得しているアプリケーションはAP1であると設定する(ST16)。またこのときAP2のアクセス対象論理デバイスが必要とするリソースの少なくとも一つ以上に対してアクセス不可となるため、AP2はアクセス対象論理デバイスにアクセス不可となる。そこでアプリケーション情報記憶部12に保持されているAP2のアプリケーション情報をアクセス不可に設定する(ST17)。またST12において、空きのリソース(アクセス権を取得しているアプリケーションが存在しないリソース)が存在する場合は、そのリソースに、アクセス権を取得しているアプリケーションはAP1であると情報を設定する。また、ST15において、AP1よりAP2の方が優先度が高い場合は、リソース情報の更新はおこなわない。またST11において同一リソースが複数存在しない場合、つまり図2や図4の例のような場合は、そのリソースに対してアクセス権を取得しているアプリケーションAP2が存在するか、リソース情報記憶部14を参照して調べる(ST18)。そして存在する場合は、アプリケーション情報記憶部12からAP2のアプリケーション情報を取得し(ST19)、ST14の処理を行う。以上ST11からST19までの処理をアクセス対象論理デバイスDEV1が必要とする全てのリソースに対して行う(ST20)。このST20の判定処理は使用リソース取得部13が行う。ST11からST19までの処理はリソースアクセス判定部15が行う。そして全てのリソースに対して処理が終了したら、使用リソース取得部13は、そのDEV1が必要とする全てのリソースIDをアクセス対象アクセス判定部16に通知し、アクセス対象アクセス判定部16は、DEV1が必要とする全てのリソースのアクセス権をAP1が取得できたかを、リソース情報記憶部14に保持されているリソース情報を参照して検索する(ST5)。そしてアクセス対象アクセス判定部16はその結果を実行部17に通知し、必要とする全てのリソースにアクセス可能な場合は、実行部17はアプリケーション情報記憶部12に保持されているAP1のアプリケーション情報にアクセス可能であると設定する(ST6)。AP1からアクセス要求がきたときは、このアプリケーション情報に設定された

アクセス可能か不可かの情報を参照して判定する。次に実行部 17 は、DEV1 のバスドライバを実行する (ST7)。また ST5 にてアクセス不可なりソースが存在する場合は、実行部 17 は、AP1 のアプリケーション情報にアクセス不可であると設定する (ST8)。

#### 【0024】

次に図 7 にアプリケーションのアクセス要求時の処理手順を示す。アプリケーションはアクセス要求をする前にアクセス開始要求をしており、そのときにアクセス対象論理デバイスへアクセス可能かどうか判定され、アクセス可能／不可の情報がアプリケーション情報に保持されているため、その情報を用いて処理を行う。まずアプリケーション IF 部 11 は、アプリケーション AP1 からアクセス対象論理デバイス DEV1 へのアクセス要求を受信する (ST21)。次にアプリケーション IF 部 11 はアプリケーション情報記憶部 12 に保持されている AP1 のアプリケーション情報を参照し、DEV1 にアクセス可能かどうかを調べる (ST22)。そして DEV1 へのアクセスが可能と設定されていた場合は (ST23)、アプリケーション IF 部 11 は実行部 17 を呼び出し、実行部 17 は、DEV1 のデバイスドライバを実行する (ST24)。また ST23 にて DEV1 にアクセス不可であると設定されていた場合は、アプリケーション IF 部 11 は、AP1 にエラーを返す (ST25)。

#### 【0025】

次に図 8 にアプリケーションのアクセス終了要求時の処理手順を示す。まずアプリケーション IF 部 11 は、アプリケーション AP1 からアクセス対象論理デバイス DEV1 へのアクセス終了要求を受信する (ST31)。次にアプリケーション IF 部 11 は、AP1 から DEV1 へのアクセス終了要求がきたことを使用リソース取得部 13 に通知し、使用リソース取得部 13 は DEV1 が必要とするリソースの ID を取得し、そのリソース ID をリソースアクセス判定部 15 に通知する。そしてリソースアクセス判定部 15 はリソース情報記憶部 14 からそのリソース情報を取得する (ST32)。次に AP1 がアクセス権を取得しているリソースに対しては、そのリソースを必要とするアクセス対象論理デバイスに対して、アクセス開始要求をしているがアクセス権を取得できていないアプリケ

ーションが存在しないか検索し、存在する場合はそのアプリケーションにリソースへのアクセス権を与える（ST33）。これはAP1がDEV1へのアクセスを終了することによって今までリソース競合が起き、アクセスが不可になっていたアプリケーションを、アプリケーションがそのことを意識せずに、つまりアプリケーションからみれば自動的にアクセス可能に設定するためである。これにより、例えばアプリケーションAP2がアクセス対象論理デバイスDEV2にアクセス開始要求をし、実際にはまだDEV2にアクセスしていないときに、AP2より優先度の高いAP1がDEV1にアクセス開始要求をしてきた場合を考える。ここで、DEV1とDEV2はともにリソースRを必要とする。始めにAP2がDEV2にアクセス開始要求をしたときは、リソースRはアクセス競合が起きていないため、AP2はDEV2へのアクセスが可能である。次にAP1がDEV1に対してアクセス開始要求をすると、AP1の方が優先度が高いため、リソースRのアクセス権はAP1が取得し、AP2はDEV2にアクセス不可となる。しかしAP2は実際にDEV2に対してアクセス要求をしていないため、アクセス不可になったことは知らない。そしてAP1がDEV1にアクセスし、必要な処理を終えてアクセス終了要求をする。そしてリソースRはAP1から解放される。DEV2はリソースRを必要としている。そこでAP2がリソースRへのアクセス権を取得できればAP2は自分がアクセス開始要求をして、実際にDEV2にアクセスするまでの間にDEV2へアクセス不可となっていたことを知る必要なく、DEV2へアクセスでき、デバイスアクセスするときのアプリケーションの作りが簡潔になる。このST33の処理の詳細なフローチャートを図9に示す。まずリソースアクセス判定部15は、AP1のアクセス対象論理デバイスDEV1が必要とするリソース情報を参照して、AP1がアクセス権を取得しているか調べる（ST41）。そして取得している場合は、リソースアクセス判定部15は、そのリソースRを使用リソース取得部13に通知し、使用リソース取得部13は、使用リソース取得部13が保持しているデバイス情報を参照してそのリソースRを必要とするアクセス対象論理デバイスを検索し、次にアプリケーション情報記憶部12が保持しているアプリケーション情報を参照して、そのアクセス対象論理デバイスにアクセス開始要求をしているアプリケーションが存在

するか調べる (ST42)。存在する場合は、使用リソース取得部13は、そのアプリケーションIDをリソースアクセス判定部15に通知する。そしてリソースアクセス判定部15は、次にそのリソースRを必要としているアプリケーションが複数存在するか判定する (ST43)。そして複数存在する場合は、リソースアクセス判定部15は、その中から最も優先度の高いアプリケーションAP3を、アプリケーション情報記憶部12が保持しているアプリケーション情報を参照して検索する (ST44)。次にリソースアクセス判定部15は、リソース情報記憶部14が保持しているリソースRのリソース情報に、アクセス権を取得しているアプリケーションはAP3であると設定する (ST45)。またST43にてリソースRを必要とするアプリケーションがアプリケーションAP3のひとつか存在しない場合は、ST45にてAP3がアクセス権を取得したと設定する (ST45)。またST43にてリソースRを必要とするアプリケーションが存在しない場合は、アクセス権を取得しているアプリケーションがないので、ST45にてリソース情報にNULLを設定する。またST41にてそもそもAP1がリソースRのアクセス権を取得していない場合はST42からST45までの処理はおこなわない。これらST41からST45までの処理を、AP1のアクセス対象であるDEV1が必要とする全てのリソースに対して行う (ST46)。このST46の判定処理は使用リソース取得部13が行う。次にST33において新たにアクセス権を取得したリソースがあるアプリケーションAP3が存在する場合、使用リソース取得部13はAP3が必要とする全てのリソースIDをアクセス対象アクセス判定部16に通知し、アクセス対象アクセス判定部16は、AP3のアクセス対象論理デバイスが必要とする全てのリソースのアクセス権を取得しているか調べる (ST34)。そしてアクセス対象アクセス判定部16はその結果を実行部17に通知し、必要とする全てのリソースのアクセス権を取得しているアプリケーションAP2が存在する場合は (ST35)、実行部17は、アプリケーション情報記憶部12に保持されているAP2のアプリケーション情報に、アクセス対象論理デバイスにアクセス可能である、と設定をする (ST36)。さらに実行部17は、デバイスドライバに対して、デバイスの設定をAP2が使用する設定に変更するように要求し、デバイスドライバは設定を

変更する (ST37)。ただしこれはAP2がアクセス開始要求をして、デバイスの初期設定をしたときに、デバイスドライバがその設定を保持しておく必要がある。もしこの機能がないデバイスドライバの場合は、AP2がアクセス要求をしてきたときにデバイス設定が必要だというエラーコードをAP2に返せばよい。またデバイスドライバがこの機能をサポートしている場合は自動復元が可能となる。ST36、ST37の処理を、ST34にて必要とする全てのリソースのアクセス権を取得した、全てのアプリケーションに対して行う。最後に実行部17はアプリケーションIF部11に対して処理が終了したことを通知し、アプリケーションIF部11は、アクセス終了要求をしてきたAP1のアプリケーション情報をアプリケーション情報記憶部12から削除する (ST38)。またST35にて必要とする全てのリソースのアクセス権を取得したアプリケーションが存在しない場合は、ST36、ST37の処理はおこなわずにAP1のアプリケーション情報を削除する (ST38)。

#### 【0026】

以上の処理の流れにより、図2の構成例の場合において具体的にどのようなかを説明する。例えば図2において、既にAP2がSDSP論理デバイスにアクセス可能であるときに、AP1がMIDI論理デバイスにアクセス開始要求をしたときを考える。AP1の方がAP2より優先度が高いとする。まずAP1が必要なMIDIリソースとスピーカリソースのリソース情報を取得する。MIDIリソース、スピーカリソースともに同一リソースは1つずつしか存在しないため、次にこれらのリソースのアクセス権を既に取得しているアプリケーションを検索する。スピーカリソースについてはAP2が既にアクセス権を取得しているため、次にAP1とAP2の優先度を比較する。そしてAP1の方が優先度が高いため、AP1はMIDIリソース、スピーカリソースともにアクセス可能となり、MIDI論理デバイスに対してアクセス可能となる。またAP2はスピーカリソースにアクセス不可となったため、SDSP論理デバイスに対してアクセス不可となる。この情報をそれぞれのアプリケーション情報に設定する。次にAP2がアクセス要求をすると、既にアプリケーション情報にアクセス不可と設定されているため、AP2にはエラーが返り、AP2はアクセス不可なこ

とを知る。次に、AP 2 がアクセス要求をする前に、AP 1 がアクセス終了要求をした場合を考える。まず AP 1 がアクセス権を取得していたリソースとして MIDI リソースとスピーカリソースを取得する。次に MIDI リソースに対してはアクセス権を必要としているアプリケーションが存在しないため、MIDI リソース情報に NULL を設定する。またスピーカリソースに対しては、AP 2 がアクセス権を必要としているため、スピーカリソース情報に、AP 2 がアクセス権を取得したと設定する。もし AP 2 以外にもスピーカリソースを必要とするアプリケーション AP 3 が存在した場合は、AP 2 と AP 3 のアプリケーションの優先度を比較し、優先度の高いアプリケーションにアクセス権を取得させる。次に AP 2 はスピーカリソースのアクセス権を新たに取得することにより、必要とするリソース全てにアクセス可能となったため、SDSP 論理デバイスにアクセス可能になり、AP 2 のアプリケーション情報にもアクセス可能であると設定される。そして SDSP デバイスドライバが AP 2 の設定に復元する機能がある場合は、次に AP 2 がアクセス要求をしたときに AP 2 はアクセス不可になっていたことを知る必要なく、SDSP 論理デバイスにアクセスできる。つまり SDSP 物理デバイスとスピーカ物理デバイスにアクセスできる。

#### 【0027】

次に図 3 の構成例の場合において具体的な処理の流れを説明する。例えば図 3 においてアプリケーション AP 1、AP 2、AP 4 が既に回線論理デバイスにアクセス可能であるときに、AP 3 が回線論理デバイスにアクセス開始要求をしてきたときを考える。優先度は  $AP 1 > AP 2 > AP 3 > AP 4$  の順で AP 1 が最も高いとする。まず回線リソースは 3 つ存在するため、その全てに対して、アクセス権を取得しているアプリケーションが設定されていないリソースがないか検索する。既に AP 1、AP 2、AP 4 の 3 つのアプリケーションがそれぞれ回線リソースのアクセス権を取得しているため、空いている回線リソースは存在しない。そこで次に AP 1、AP 2、AP 4 の中で最も優先度の低いアプリケーションを検索する。AP 4 が最も優先度が低い。次に AP 4 と AP 3 の優先度を比較し、AP 3 の方が優先度が高いため、AP 4 がアクセス権を取得していた回線リソースに対して、AP 3 がアクセス権を取得する。そして AP 3 は回線論理デバ

イスにアクセス可能となる。またAP4はアクセス不可となる。次にAP4より優先度の低いAP5がアクセス開始要求をしてきた場合を考える。既に回線リソースは全て埋まっているため、AP1、AP2、AP3の中で優先度の最も低いアプリケーションを検索する。次に検索結果のAP3とAP5の優先度を比較し、AP5の方が優先度が低いため、AP5はアクセス不可となる。次にAP4、AP5がともに回線論理デバイスにアクセス要求をする前に、AP2がアクセス終了要求をした場合を考える。回線リソースを必要としているアプリケーションとしてAP4、AP5の2つが存在するため、AP4とAP5の優先度を比較する。そして優先度の高いAP4が、AP2がアクセス権を取得していた回線リソースのアクセス権を取得する。そしてAP4は回線論理デバイスにアクセス可能となる。

#### 【0028】

次に図4の構成例の場合において具体的な処理の流れを説明する。例えば図4においてアプリケーションAP1が既にSDSP録音論理デバイスにアクセス可能であるときに、AP3がSDSP再生論理デバイスにアクセス開始要求をしたときを考える。AP1の方がAP3より優先度が高いとする。またAP1とAP3は同じコーデックXを使用するとする。またSDSPコーデックリソースは付加情報を持ったリソースで、設定された状態と同じであれば多重アクセスを許す性質を持っている。まずSDSPコーデックリソースは既にアクセス権を取得しているAP1が存在するため、AP1とAP3の優先度を比較する。AP1の方が優先度が高いため、普通のリソースであればAP3はアクセス権を取得できないが、SDSPコーデックリソースは多重アクセスを許すリソースであるため、SDSPコーデックリソースに設定されているコーデックの種別とAP3の使用するコーデックの種別を比較する。そして同じコーデックXであるため、AP3もSDSPコーデックリソースへのアクセス権を取得し、SDSP再生論理デバイスに対してアクセス可能となる。次にAP2がSDSP録音論理デバイスにアクセス要求をした場合を考える。AP2の優先度はAP1とAP3の間である。まずSDSPコーデックリソースに既にアクセス可能となっている最も優先度の高いアプリケーションAP1と優先度を比較する。AP1の方が優先度が高いた

め、次にSDSPコーデックリソースに設定されているコーデックの種別とAP2の使用するコーデックの種別を比較する。ここでAP2はAP1やAP3とは異なるコーデックYを使用する場合、AP2はSDSPコーデックリソースのアクセス権を取得できないことになり、SDSP録音論理デバイスにはアクセス不可となる。次にAP1がアクセス終了要求をしたときを考える。まずSDSP録音リソースに対しては、AP2が必要としているため、AP2にアクセス権を取得させる。次にSDSPコーデックリソースに対しては、AP2とAP3が必要としているため、AP2とAP3の優先度を比較する。そしてAP2の方が優先度が高いため、AP2がSDSPコーデックリソースのアクセス権を取得する。またコーデックの種別を設定しなおす。次にSDSPコーデックリソースに新たに設定されたコーデックの種別とAP3が使用するコーデックの種別を比較する。ここでAP2とAP3は異なるコーデックを使用するため、AP3はSDSPコーデックリソースにアクセス不可となり、SDSP再生論理デバイスに対してアクセス不可となる。

#### 【0029】

##### 【発明の効果】

以上の説明のとおり本発明によれば、アプリケーションのアクセス対象のデバイスと実際に存在する物理デバイスと競合調停を行う対象のリソースとを分離して考えることで、デバイスの構成やデバイスの特性に合わせたリソース構成をとることができ、例えばアクセス数がある数までは多重アクセスを許す競合調停や、ある条件を満たせば多重アクセスを許す競合調停など、複雑な競合調停が実現可能となる。また例えば他のデバイスと配線結合され、単体ではI/Oポートを持たないデバイスが複数のデバイス間で共有されるような場合でも、そのデバイスへのアクセス競合調停が実現可能となる。

#### 【0030】

また例えば複数の物理デバイスを同時に制御することでアプリケーションの処理が実行される場合において、同時アクセスが必要な全ての物理デバイスにアクセス可能な場合は該当のデバイスドライバが呼び出され処理が実行されるが、一つでもアクセス不可なデバイスがある場合はアプリケーションにエラーが戻ると



いう仕組みが必要である。この場合に、アプリケーションのアクセス対象デバイスに対応するリソースを、同時にアクセスが必要な複数の物理デバイスとする構成をとることでこの仕組みを実現できる。

#### 【0031】

またアプリケーションに付与した優先度を用いて競合調停を行うことで、例えば多重アクセスを許すデバイスにおいては、優先度のより高いものからアクセスが可能となる、といった優先度に応じた複雑な排他制御をすることが可能となる。

#### 【0032】

またアプリケーションがデバイスにアクセス可能かどうかをアプリケーション情報に持たせ、デバイスへアクセス要求をしているアプリケーションの構成が変更するときにこのアプリケーション情報を更新することで、アプリケーションがデバイスへのアクセス要求をする度にリソースアクセス判定部およびアクセス対象アクセス判定部による競合調停をする必要がなくなり、処理の高速化が図れる。さらにアプリケーションがアクセス要求したときに、アクセスが可能であれば該当のデバイスドライバが呼び出され処理し、不可であればアプリケーションにエラーが戻るという仕組みにより、アプリケーションはアクセス要求をするときにエラーの場合の対処をするだけでよく、アプリケーション開発者の負担が軽減される。さらに、他のアプリケーションのアクセス要求により、デバイスへのアクセス権を奪われたアプリケーションが次にアクセスするまでに再びそのデバイスへのアクセスが可能になった場合、アプリケーションは一度アクセスが不可になったということを知ることなく、特別な処理をせずにデバイスへのアクセスをすることができる。

#### 【図面の簡単な説明】

##### 【図1】

本発明の実施の形態の構成例の概要を示す図

##### 【図2】

アクセス対象論理デバイスとリソースと物理デバイスとの構成例を示す図

##### 【図3】

アクセス対象論理デバイスとリソースと物理デバイスとの構成例を示す図

【図 4】

アクセス対象論理デバイスとリソースと物理デバイスとの構成例を示す図

【図 5】

アクセス開始要求時の処理手順を示すフローチャート

【図 6】

アクセス開始要求時の詳細な処理手順を示すフローチャート

【図 7】

アクセス要求時の処理手順を示すフローチャート

【図 8】

アクセス終了要求時の処理手順を示すフローチャート

【図 9】

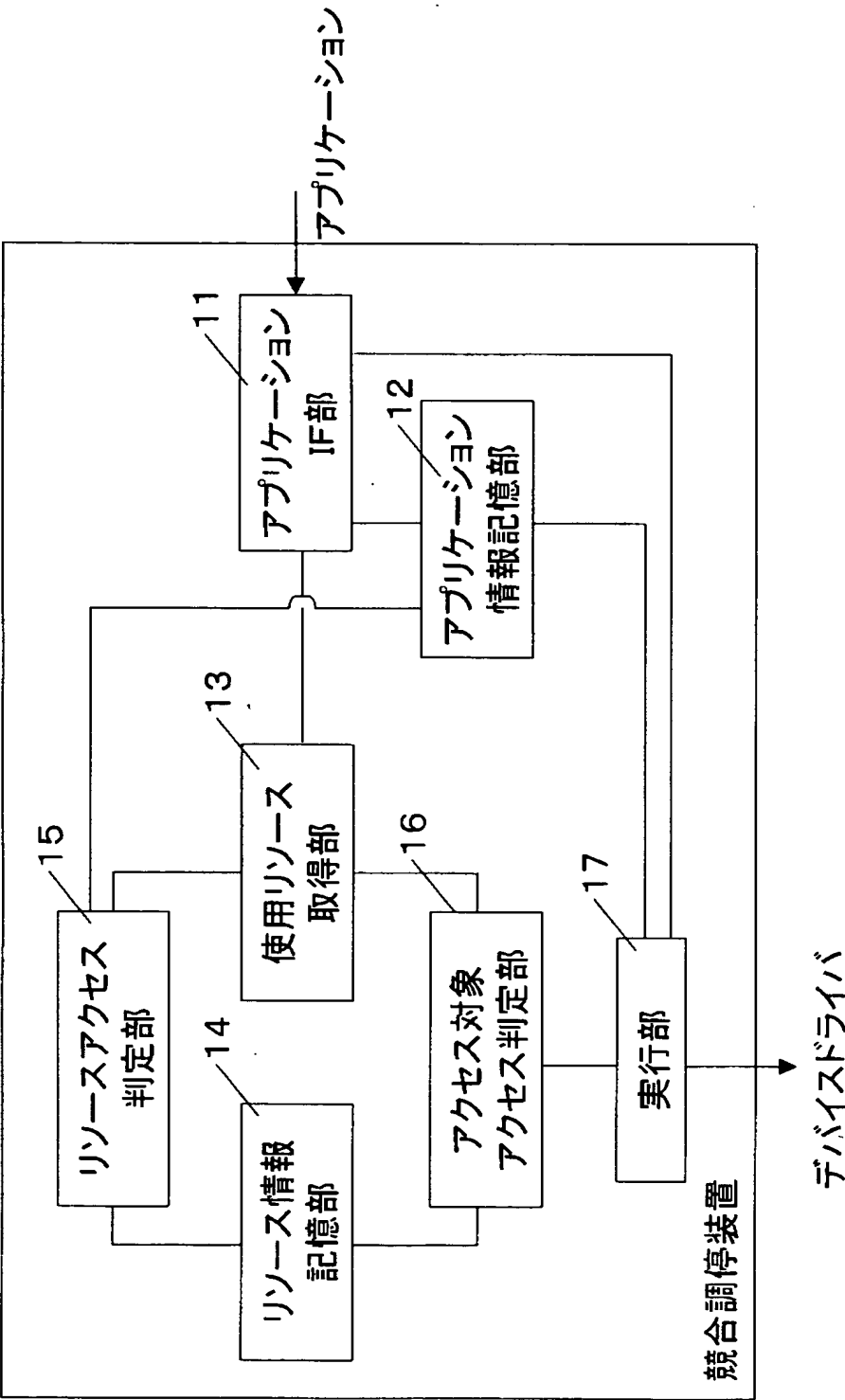
アクセス終了要求時の詳細な処理手順を示すフローチャート

【符号の説明】

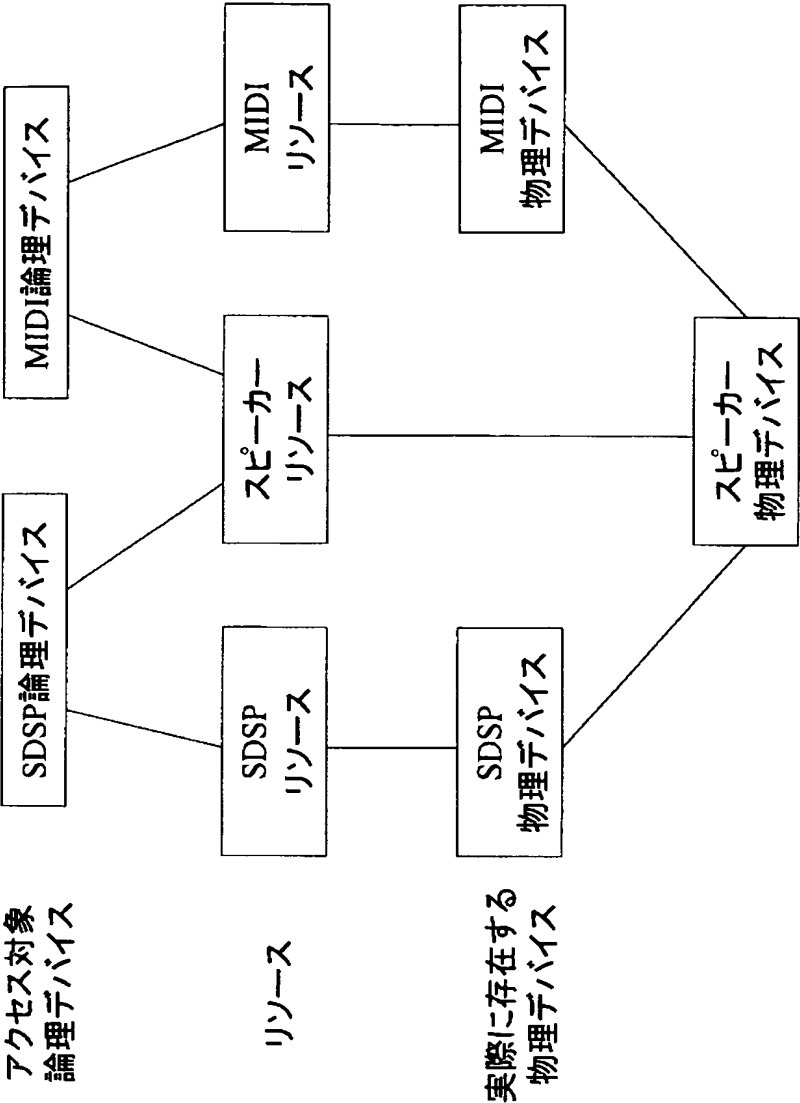
- 1 1 アプリケーション I F 部
- 1 2 アプリケーション情報記憶部
- 1 3 使用リソース取得部
- 1 4 リソース情報記憶部
- 1 5 リソースアクセス判定部
- 1 6 アクセス対象アクセス判定部
- 1 7 実行部
- 3 2 1 つ目の回線リソース
- 3 3 2 つ目の回線リソース
- 3 4 3 つ目の回線リソース

【書類名】 図面

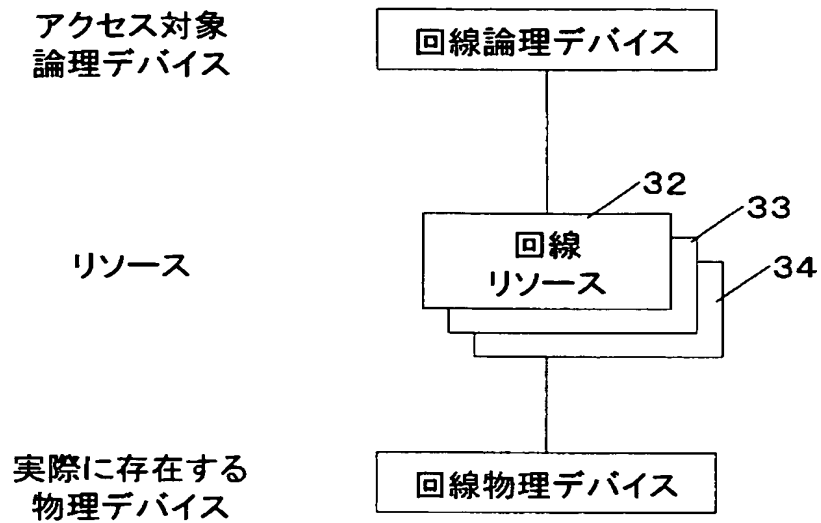
【図 1】



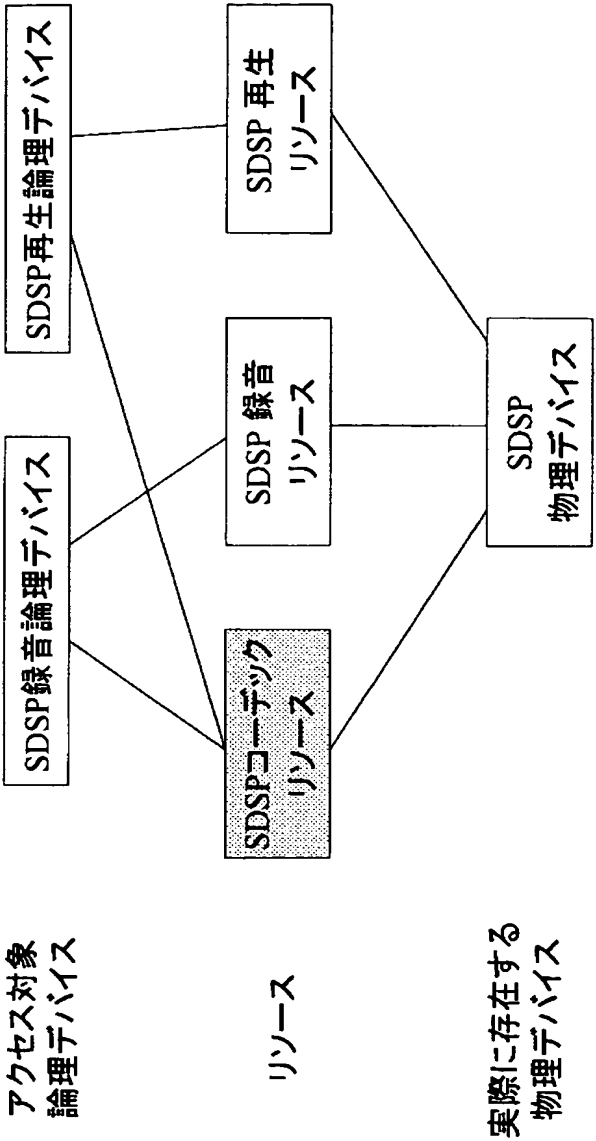
【図 2】



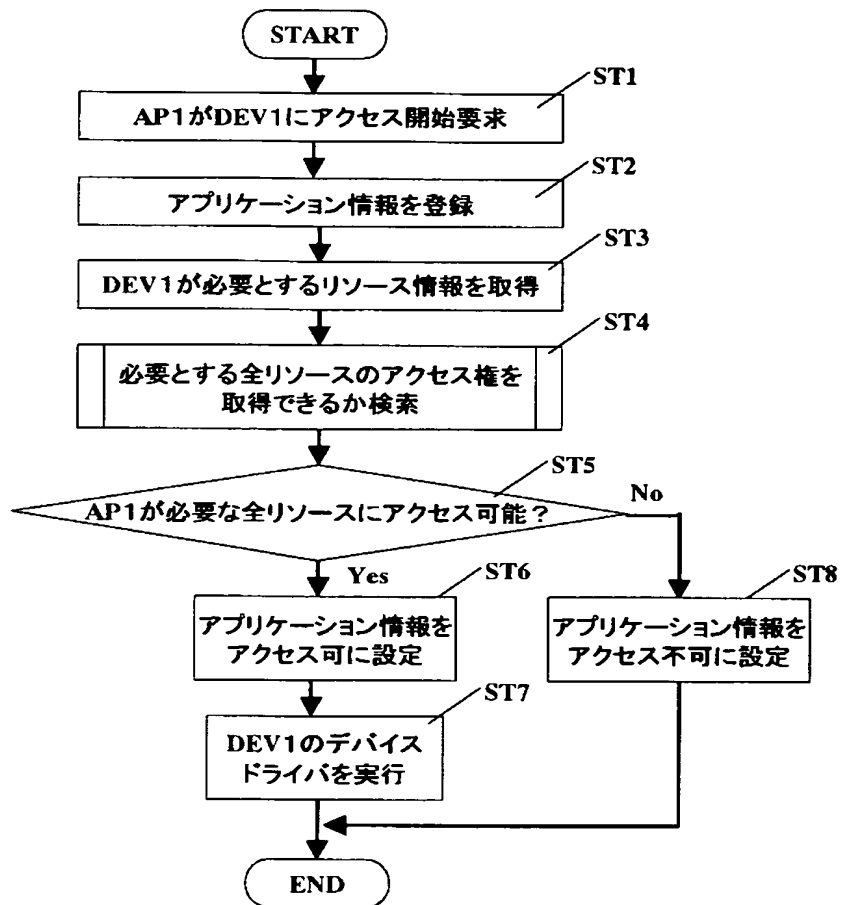
【図 3】



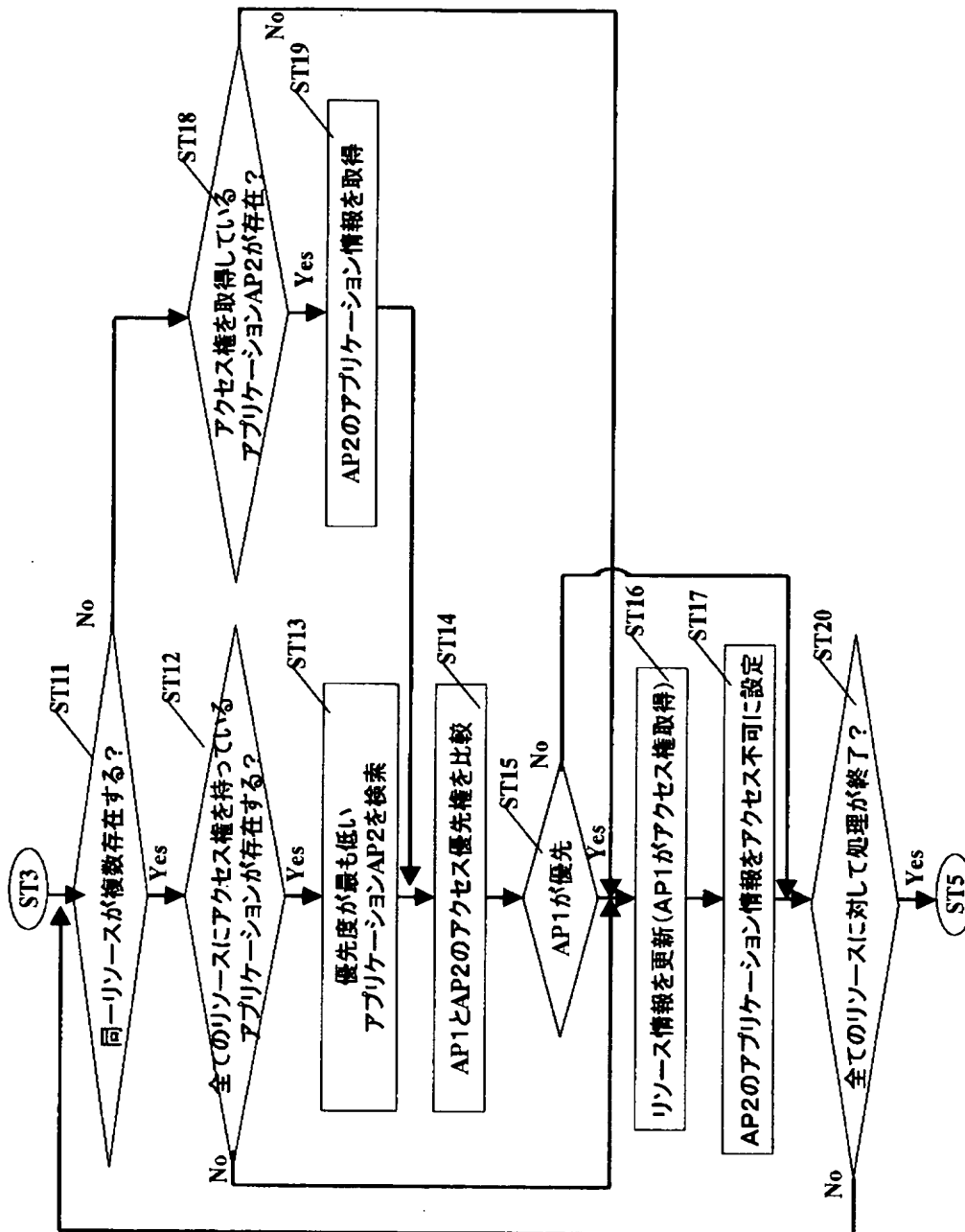
【図 4】



【図 5】

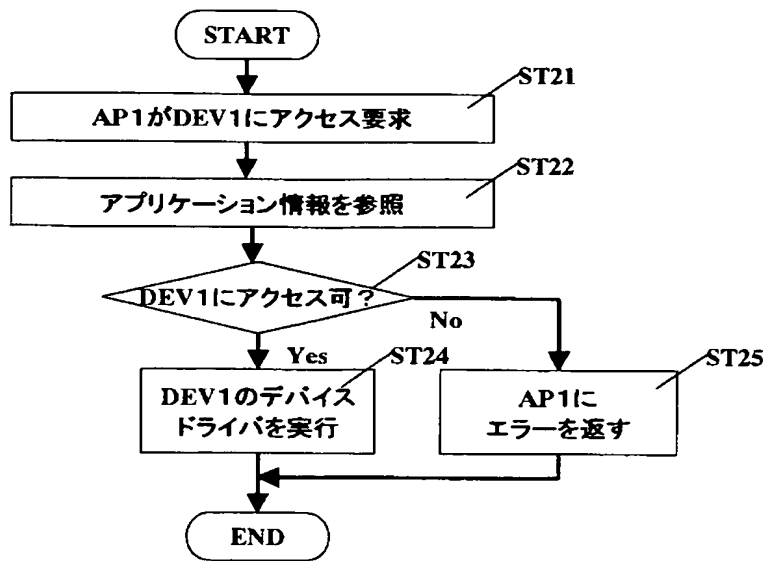


【図 6】

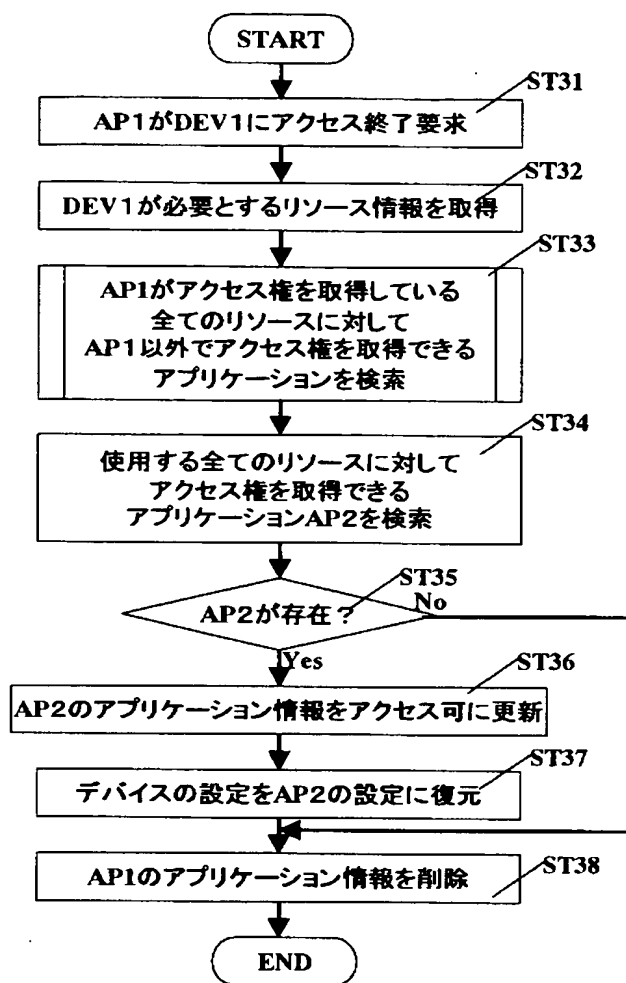




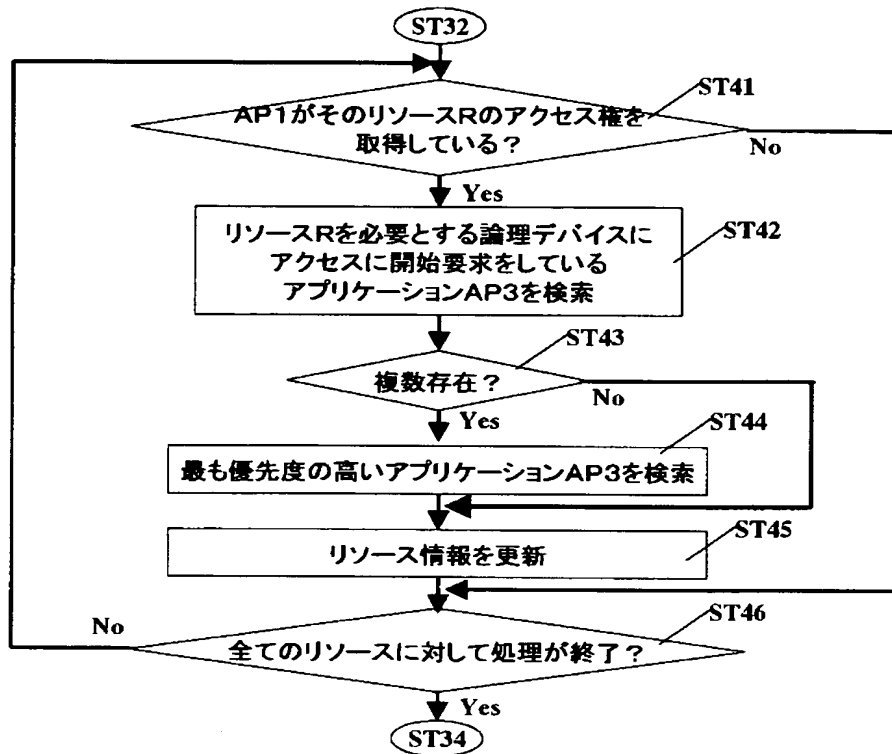
【図 7】



【図 8】



【図 9】



【書類名】 要約書

【要約】

【課題】 アプリケーションのアクセス対象であるデバイスと競合調停を行う対象とを分離することで複雑な競合調停を可能にすることを目的としている。

【解決手段】 アプリケーションからアクセス要求を受信するアプリケーション I F 部と、アクセス対象のデバイスが使用するリソースを取得する使用リソース取得部と、そのリソースに対して現在アクセス権を持っているアプリケーションとアクセス要求をしているアプリケーションのどちらにアクセス権を持たせるか判定するリソースアクセス判定部と、アプリケーションがアクセス対象にアクセス可能かを判定するアクセス対象アクセス判定部と、デバイスを制御するデバイスドライバを呼び出す実行部とを備えた競合調停装置。

【選択図】 図 1

特願 2 0 0 2 - 3 3 1 9 0 5

出 願 人 履 歴 情 報

識別番号

[ 0 0 0 0 0 5 8 2 1 ]

1. 変更年月日

1 9 9 0 年 8 月 2 8 日

[変更理由]

新規登録

住 所

大阪府門真市大字門真 1 0 0 6 番地

氏 名

松下電器産業株式会社